Australian Government
Department of Defence
Defence Science and
Technology Organisation

Development of a Rotary Wing Unmanned Aerial Vehicle (UAV) Simulation Model

Matthew Reid and Sylvain Manso

Aerospace Division
Defence Science and Technology Organisation

DSTO-GD-0791

ABSTRACT

This report details the work undertaken during a DSTO Summer Vacation Scholarship Program on the development of a flight simulation system for the Raptor 90 UAV platform. Simulink was used to execute the core simulation update loop, with externally linked modules providing functionality for joystick input, flight dynamics, and 3D visualisation. A flight model for the Raptor was developed in FLIGHTLAB, and flown in the simulation system under pilot control and autonomous control. The Raptor model was shown to be capable of flying autonomously through a range of manoeuvres, via the use of an autopilot code module written in MATLAB. Additionally, a MATLAB interface for an inertial motion unit was developed to aid in future autopilot research.

Published by

Aerospace Division
DSTO Defence Science and Technology Organisation
506 Lorimer St
Fishermans Bend, Victoria 3207 Australia

Telephone: 1300 333 362
Fax: (03) 9626 7999

AR-015-877
March 2014

Development of a Rotary Wing Unmanned Aerial Vehicle (UAV) Simulation Model

Executive Summary

This report details the work undertaken during a DSTO Summer Vacation Scholarship (SVS) Program. The work includes modelling of the Raptor 90 SE helicopter, a radio controlled vehicle, within the FLIGHTLAB software environment, development of a MATLAB/Simulink visualisation tool using Flightgear, and development of a Joystick and Crossbow Inertial Measurement Unit (IMU) interface.

The accuracy of the Raptor flight model is currently unknown. The model's performance and handling during flight testing appears to be qualitatively appropriate. The accuracy of the flight model would be improved significantly if experimental data from flight testing of the physical aircraft becomes available.

The real time simulation system allowed the Raptor flight model to be flown successfully under pilot control and autonomous control. The system's autopilot module proved to be capable in flying the helicopter along a predetermined flight path. Although the control system component of the autopilot was found to produce stable flight, the system suffers from minor oscillation and steady state error in velocity and heading control. These problems may be resolved in the future through the implementation of higher fidelity controllers, possibly utilising Kalman filtering to compensate for cross-coupling effects.

The IMU interface developed for MATLAB/Simulink was found to function as expected, allowing state data from the IMU to be visualised in Flightgear via Simulink. An appropriate test environment is required before IMU position and velocity can be visualised.

Table of Contents

1. Introduction

1.1 Overview

This report details the work undertaken during a DSTO Summer Vacation Scholarship (SVS) Program. The work includes modelling of the Raptor 90 SE helicopter within the FLIGHTLAB software environment, development of a MATLAB/Simulink visualisation tool using Flightgear, and development of a Joystick and Crossbow Inertial Measurement Unit (IMU) interface.

By developing an ability to simulate a Rotary Wing Unmanned Aerial Vehicle (RW-UAV), DSTO can enhance its role in advising the ADF on emerging RW-UAV technologies and their associated issues. To this end, DSTO purchased a Raptor 90 SE radio-controlled helicopter to act as a UAV avionics research platform, and to provide a means of validating RW-UAV simulation techniques.

RW-UAVs combine the advantages of a helicopter with those of conventional UAVs to provide unique capabilities. RW-UAVs can be launched and recovered from ships; they can remain stationary relative to fixed targets; and they can operate at low level near obstacles such as trees or buildings.

1.2 Raptor 90SE Helicopter

The Raptor 90 SE (Figure 1) is a Remote Control (RC) hobbyist helicopter produced by Thunder Tiger. It's the largest helicopter in the Raptor series and features a 4 kg payload capacity, making it suitable as a UAV avionics platform [1].

Raptor 90 SE specifications [2]:

- Fuselage length: 1410 mm
- Fuselage width: 190 mm
- Total height: 465 mm
- Main rotor diameter: 1640 mm
- Tail rotor diameter: 260 mm
- Full equipped weight: 4800 g

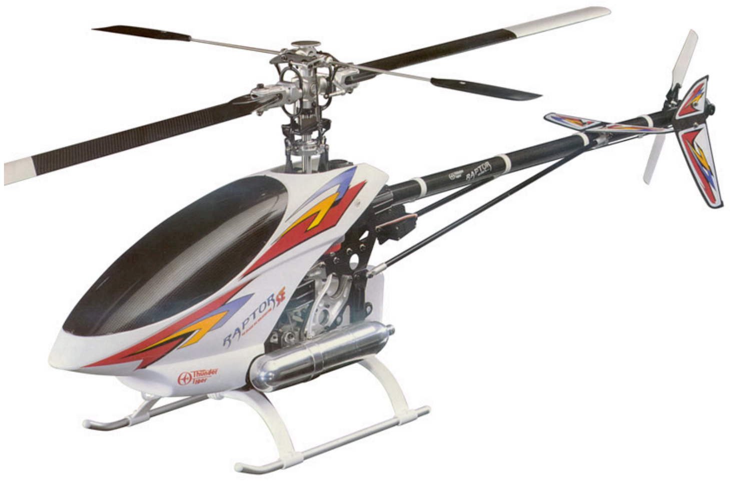

Figure 1. The Raptor 90 SE

1.3 RW-UAV Simulation

A real time simulation of an autonomous Raptor 90 SE was developed. The simulation system consists of several modules for flight dynamics, pilot control input, 3D visualisation, and autonomous control. Each of these modules is interconnected to allow the update loop for a typical UAV system to be simulated (see Figure 2).

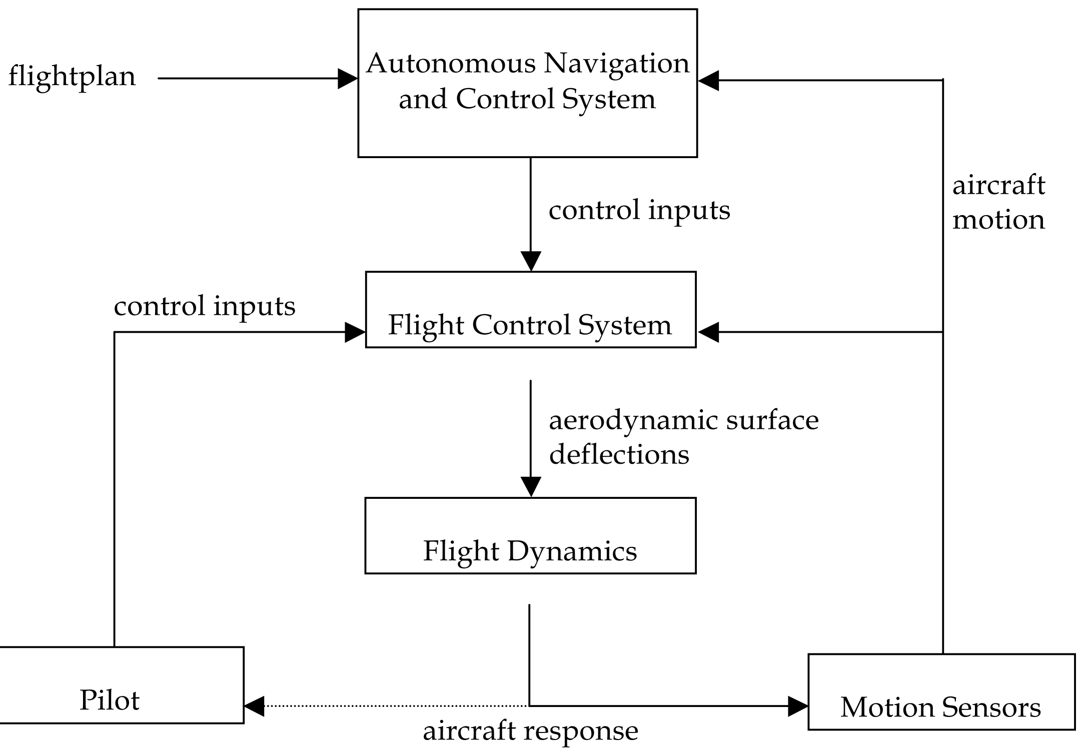

Figure 2. Update loop for typical UAV

2. Flight Model

2.1 Overview

The flight model for the Raptor was developed in FLIGHTLAB. Properties relating to the physical structure of the helicopter were measured from the physical aircraft and inserted into the flight model. Due to a lack of available performance data, approximations and 'best guesses' were used for many of the higher fidelity model parameters. It is anticipated that the flight model will be refined in the future once data from Raptor flight tests becomes available.

2.2 Main Rotor

Blade Model

Initially, the FLIGHTLAB Blade Element model was used to simulate the main rotor. Due to computational stability problems with this model however, the simpler Disk Main Rotor model was used instead. It is assumed that since the Raptor is relatively small in size, the Blade Element solver was unable to converge upon a solution due to the small dimensions used in the model. This issue is not seen in the full-scale helicopter FLIGHTLAB models. Although the Disk Main Rotor is less physically accurate than the Blade Element model, it has a benefit in that it requires fewer aerodynamic properties to be known about the aircraft. Since the availability of aerodynamic and performance data for the Raptor is limited, the use of the Disk Main Rotor module allows the FLIGHTLAB model to be developed more easily.

Rotor Direction

Although the Raptor has a clockwise rotor (when viewed from above), a counter-clockwise rotor was used in the flight model due to float overload errors generated when trimming the clockwise rotor in FLIGHTLAB. The source of this problem is unknown and warrants further investigation. This issue is not seen in the full-scale helicopter FLIGHTLAB models at DSTO. Due to the change in rotor direction, the tail rotor in the flight model was placed on the left side of the helicopter rather than the right, in order for the flight model to be correctly mirrored. As a result of this mirroring effect, the directions of yaw and heave coupling will most likely be inverted, as will the direction of lateral drift from tail rotor thrust when compared with the actual helicopter.

Blade Twist

From inspection of the Raptor's main rotor blades, there appears to be little or no blade twist. Unfortunately FLIGHTLAB generated a float overload error when setting the Linear Blade Twist property to zero. This problem was solved by setting the value to 0.1°.

Rotor RPM

The main rotor RPM for the raptor is stated as being approximately 1500RPM for nominal conditions, and between 1800-1900RPM for aerobatics. The lower value of 1500RPM was chosen as a more realistic representation of performance in the flight envelope where the UAV will typically operate.

Swashplate Phase Angle

The linkage that controls rotor blade pitch is offset from the pitch axis of rotation of the blade (see Figure 3). This leads to a phase angle lag (or lead, depending on the hub design) between the blade angle and the swashplate angle at any given rotor azimuth. This phase angle is commonly referred to as swashplate phase angle. It is unknown whether the Raptor control system automatically compensates for phase angle coupling or whether this must be handled by the pilot, however it was assumed that the control system is responsible for some degree of decoupling in order to decrease pilot workload. Through testing the control response of the FLIGHTLAB model, a value of -15º was chosen for the phase angle offset of the swashplate, as this appears to produce minimum cross-coupling in pitch and roll response.

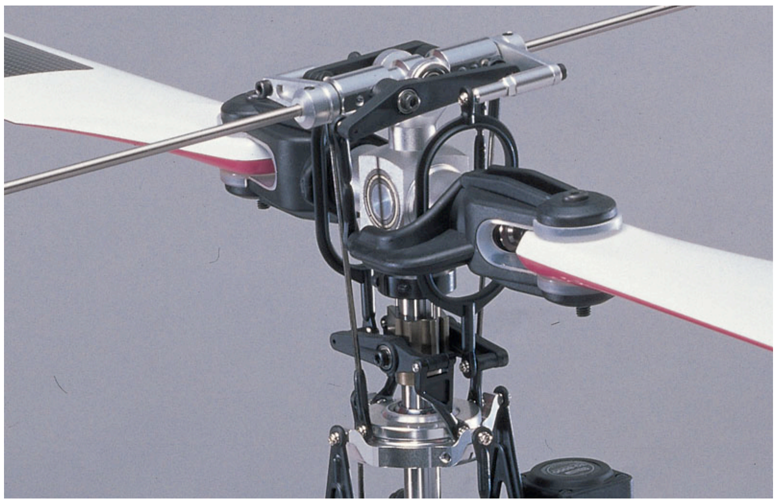

Figure 3. The Raptor hub and swashplate

Aerodynamic Properties

Since no experimental data was available on rotor aerodynamic parameters, default values for the FLIGHTLAB software were used. Since these parameters are non dimensional, it is assumed that the default values will scale correctly and provide an adequate approximation for the Raptor.

Blade Flapping

The Raptor's blade flapping moment of inertia was calculated by approximating the blade with a rectangular prism with the axis of rotation at one end, according to the following equation,

$$I = \frac{MR^2}{3}$$

where M is the blade mass and R is the blade radius.

The Disk Rotor model requires a blade flap hinge offset to be specified. Since the Raptor rotor is hingeless, a hinge offset value at 33% along the rotor was arbitrarily chosen as a representative mean value. The blade flapping frequency was left at the FLIGHTLAB default value due to a lack of available data.

2.3 Tail Rotor

The Bailey Tail Rotor model was used due to its simplicity, given the lack of data available. This rotor model represents a disk rotor with a non-feathering blade assumption. Thus, there is only collective pitch and no cyclic pitch input.

Linear Blade Twist

As with the main rotor blades, the Raptor has no observable tail rotor blade twist. This property was again set to 0.1° in order to overcome the FLIGHTLAB float overload error.

Rotor RPM

The gearing ratio between the main rotor and tail rotor was determined from the physical aircraft by measuring the ratio between blade turn count for each rotor. This ratio was multiplied by the main rotor nominal RPM to determine the tail rotor RPM.

Aerodynamic Properties

As with the main rotor, the default FLIGHTLAB aerodynamic properties were used for the tail rotor due to lack of available data.

2.4 Fuselage

Airloads

A generic fuselage airload table was used to approximate the Raptor fuselage, consisting of non-dimensional force coefficients for varying angles of attack and sideslip. The Raptor's characteristic length and area were chosen as overall fuselage length and approximate frontal plan area respectively.

Moments of inertia

Moments of inertia about the x, y and z axes for the Raptor 50 have previously been calculated by Frye et al using geometric primitives to approximate the Raptor 's mass distribution [3]. Since the Raptor 50 is of similar mass and length to the Raptor 90 SE, these values were used in the FLIGHTLAB model. A simple validation was performed on the Raptor 50 values by determining the radius of gyration in each axis. The gyration radii were found to lie within the fuselage at plausible locations.

The helicopter was assumed to be symmetric about the X-Z plane, making the I_{xy} and I_{yz} cross product moments equal to zero. As no existing empirical data could be obtained for I_{xz}, and due to its difficulty to measure, it was assumed to be zero also. The effect of I_{xz} on cross-coupling is expected to be negligible when compared with aerodynamic cross-coupling effects.

2.5 Propulsion

Initially the Simple Engine model was used in FLIGHTLAB, as it best approximates the small internal combustion engine used by the Raptor. The engine time constants and fuel flow parameters were approximated by scaling the default FLIGHTLAB values appropriately. Unfortunately the simple engine model was found to make the simulation unstable, despite tweaking of the engine parameters. Because of this the Ideal Engine model was used instead.

Engine torque was estimated based on available power and RPM specification data, according to the equation,

$$\tau = \frac{\dot{W}}{\omega}$$

where τ is torque, $\dot{W}$ is engine power, and ω is engine angular velocity.

The engine's power output was assumed to be constant at the manufacturer specified value. Realistically, available power falls off with altitude due to the decreased air density, however this effect is deemed negligible since the Raptor is only expected to operate close to sea level.

2.6 Landing Gear

The Raptor helicopter rests on two landing skids. Since FLIGHTLAB is only capable of modelling undercarriage wheels, the skids were approximated by four small wheels placed at the four extreme ground contact points. The strut spring parameters were scaled down from the FLIGHTLAB defaults based on the Raptor's mass and approximate skid compression amount.

3. Control System

3.1 Overview

The control system used in the Raptor FLIGHTLAB model was based on an existing system developed for the Yamaha RMAX Helicopter FLIGHTLAB model[1]. The flight control system consists of a simple linkage between the control inputs and the corresponding blade pitch or swashplate angle, with appropriate gains and biases applied. Each output component of the control system is passed through a short period lag filter to remove high frequency noise in the signal.

A stability augmentation system (SAS) was also implemented in the form of 1st order lagged rate feedback. Although helicopters are unstable by nature, a SAS system is able to decrease the rate at which a helicopter will deviate from trim, improving controllability.

3.2 Pitch and Roll Control

As with many other scale RC helicopters, the Raptor is fitted with a flybar system to improve pitch and roll stability. The flybar consists of a small set of blades on the main rotor hub which are physically linked to the swashplate and main rotor blades. The stabilizing effect of a flybar may be modeled as a 1st order lagged rate feedback system [4], as shown in Figure 4.

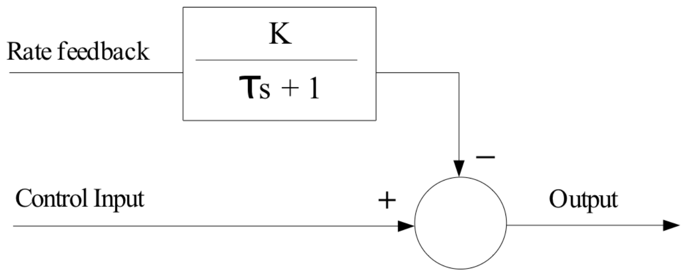

Figure 4. Flybar modelled as 1st order lagged rate feedback system

The Raptor's flybar lag time constant, , was determined analytically using the following equation [4],

$$\tau = \frac{16}{\gamma\Omega}$$

where Ω is the main rotor speed, and γ is the Lock number describing the ratio between aerodynamic and inertial forces acting on the flybar.

[1] Developed by S. Gardiner for DSTO as part of an SVS program

The Lock number is given by [4],

$$\gamma = \frac{\rho ac\left(R^4 - r^4\right)}{I_b}$$

where ρ is the air density, c is the flybar chord length, a is the flybar lift curve slope, r^4 and R^4 are the flybar inner and outer radii respectively, and I_b is the flybar moment of inertia.

Using this analytical approach, the lag time constant of the flybar was calculated to be 0.0502. Although the accuracy of this result was unable to be verified, past experiments with the Yamaha RMAX RC helicopter by Mettler show a good correlation between time constants obtained analytically and experimentally. Nevertheless, experimental testing or more complex modelling of the Raptor rotor system would be required to verify the flybar time constant. Figures 5 and 6 outline the pitch and roll control system of the Raptor FLIGHTLAB model.

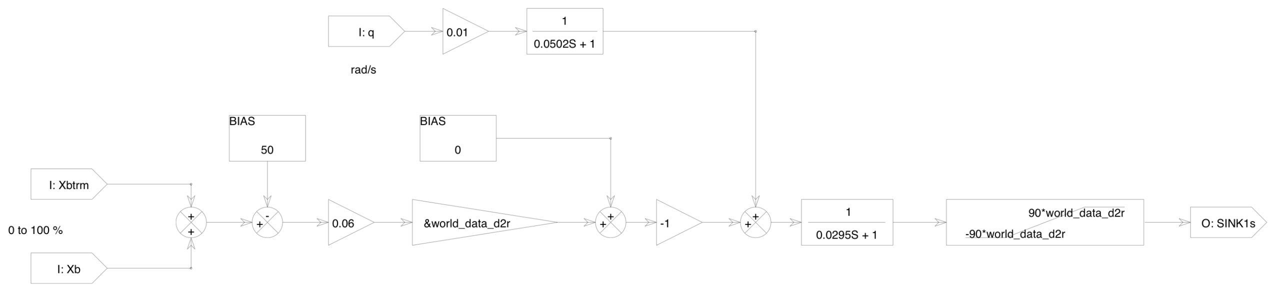

Figure 5. Pitch Control System Diagram

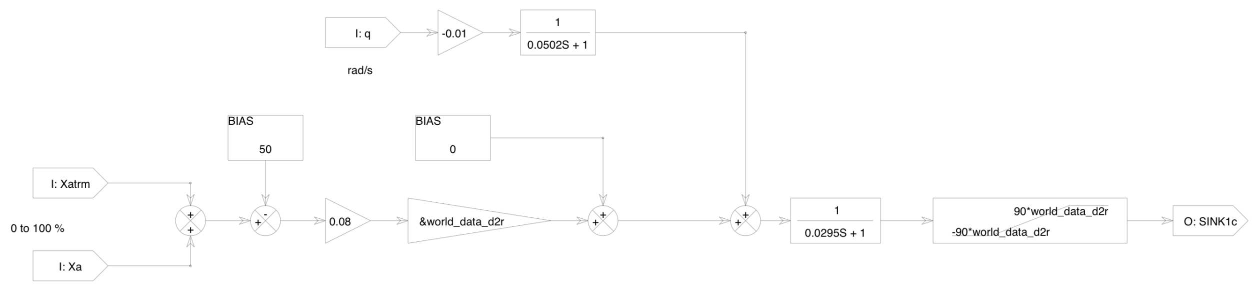

Figure 6. Roll Control System Diagram

3.3 Yaw Control

Like most small scale RC helicopters, the Raptor is fitted with a piezoelectric gyroscope to improve yaw stability. This was modelled as lagged yaw rate feedback system (see Figure 7).

The Raptor controller has a yaw-heave mixer, which passes a percentage of the collective input into the yaw control system. This has the effect of augmenting pedal input to automatically compensate for yaw induced by a change in collective. This is beneficial in that the pedal input required to trim the helicopter will be significantly less for a given collective position.

3.4 Heave Control

The heave component is the simplest part of the control system, consisting of a simple gain, bias, and noise filter. Unlike the pitch, roll and yaw components, no SAS was implemented (see Figure 8).

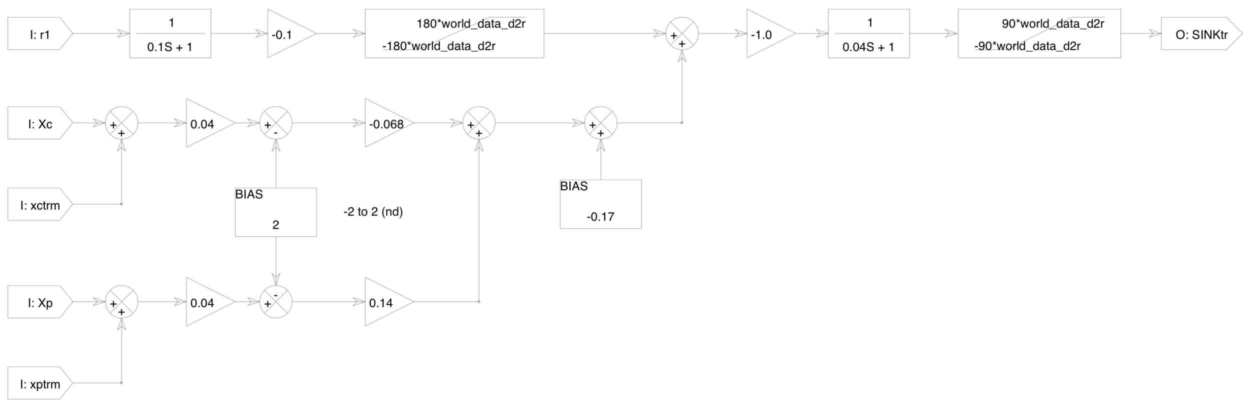

Figure 7. Yaw Control System Diagram

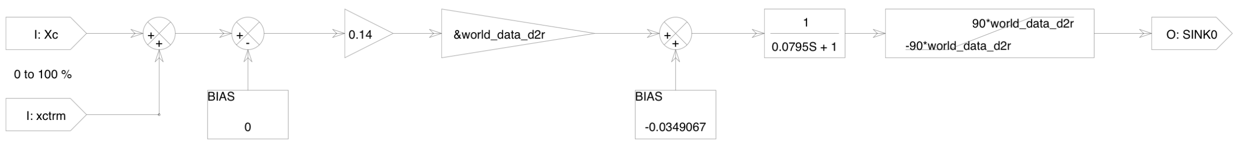

Figure 8. Heave Control System Diagram

4. Flight Performance Testing

4.1 Trim

The first step in the validation process is to ensure that the flight model is capable of reaching trim conditions; that is where the control inputs are such that no linear or angular acceleration is produced. When trimmed, it is also important that the helicopter's angular velocity is near-zero to achieve maximum stability.

When trimmed in a hover, it is desirable for the cyclic and pedal inputs to be mid-range at 50%, and for the collective to be at about 70% in order to obtain a suitable range of motion for the controls.

To analyse the Raptor flight model in trim, FLIGHTLAB's X-Analysis program was used. During the trimming process, the program attempts to vary the cyclic, collective and pedal control inputs in such a way as to reach a set of corresponding trim targets; in this case zero linear and angular accelerations. If no trim solution can be found within the upper and lower limits of the control inputs, some adjustment to the model is required.

Trimming of the Raptor was initially attempted using default control system gains and biases from the FLIGHTLAB RMAX helicopter model developed by Gardiner. No trim solution could be obtained however, most likely due to the difference in scale between the two helicopters (a ratio of 2:1 between the larger RMAX and the Raptor). The gains and biases were subsequently adjusted repeatedly until trim convergence could be obtained with mid-range control inputs (see figures 5 to 8).

4.2 Maximum Airspeed

Since no flight performance data for the Raptor was available, the approximate maximum airspeed of the helicopter was obtained from unverified speed reports on internet forums (see Appendix B for data sources). In most cases aircraft speed was reportedly measured using either a police radar gun or a GPS receiver.

From these results, an approximate top speed of 60 knots was assumed for the Raptor model. The retreating blade stall speed of the Raptor was estimated to be 250 knots, indicating that the limiting factor for top speed is most likely fuselage drag rather than retreating blade stall.

Using X-Analysis, the maximum speed of the Raptor flight model was initially found to be greater than 60 knots. In order to adjust the maximum speed, the fuselage reference area, initially estimated to be 0.0336 m^2, was increased to 0.057m^2 causing more drag to be generated. This value was obtained by using X-Analysis to generate plots of airspeed versus control inputs for different reference areas. The chosen reference area corresponded to the case where maximum collective input was required to maintain level flight at 60 knots. The trimmed control positions for forward airspeed are shown in Figure 9.

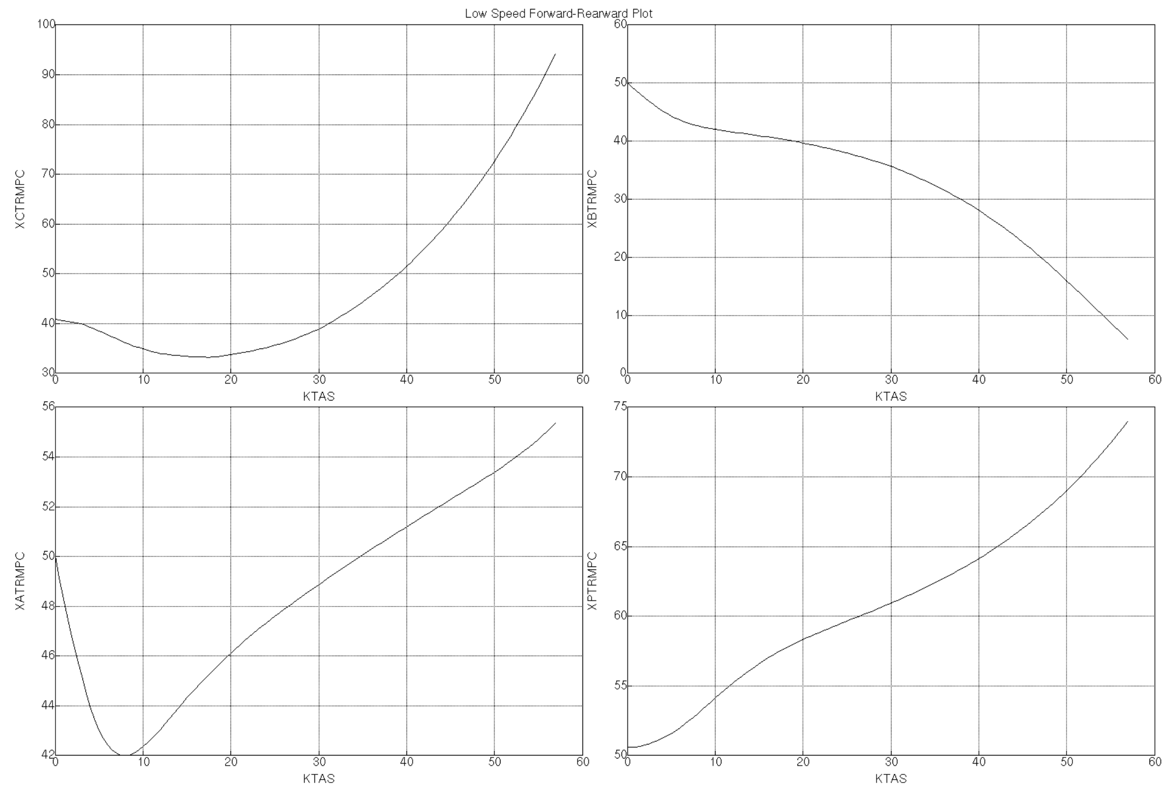

Figure 9. Plot of trimmed control inputs vs forward airspeed. Lateral (XATRMPC), Longitudinal (XBTRMPC), Collective (XCTRMPC), and Pedal (XPTRMPC) control positions in percentage are plotted against forward airspeed in knots.

4.3 Handling Qualities

Response Polarity

The first stage in validating the handling qualities of the flight model was to ensure that a control input of a certain sign produced a response of the corresponding sign. For example, a positive longitudinal cyclic input from trim is expected to produce a positive pitching response. The Nonlinear Response tool of X-Analysis was used to create plots of angular and vertical acceleration over time for a set of control step inputs. All inputs were found to generate a response with the correct sign. Had any reversed control inputs been identified, the linkage gain for that input would require a change in sign.

SAS Gains

The stability of the flight model in pitch, roll, yaw and climb was plotted with respect to time under neutral control inputs. The rate feedback gains in the SAS system were then adjusted to minimise the magnitude of long term oscillations. The model was observed to be quite stable in the short term, with oscillations only becoming noticeable after about 10 seconds (see Figure 10). Since helicopters are generally unstable in the long term, this oscillatory behaviour is expected.

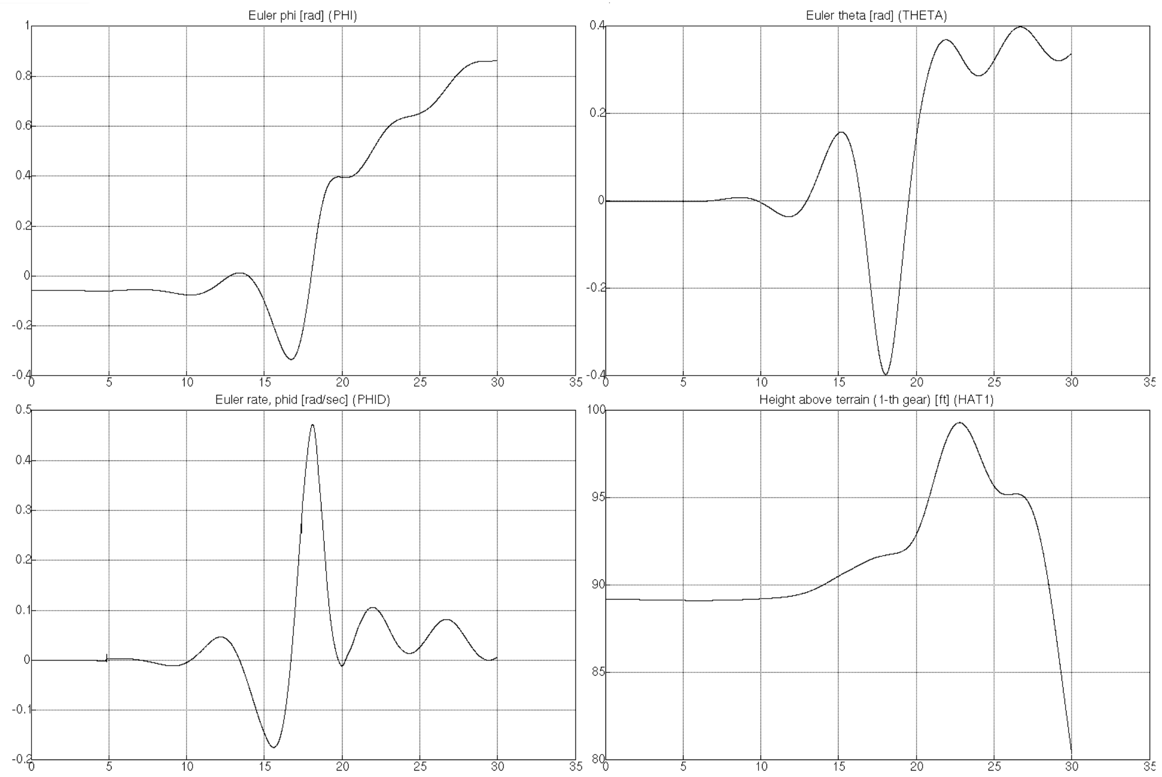

Figure 10. Long period deviation from trimmed hover with no control inputs

The response of each control input was then tested individually, both with positive and negative deflections from trim. The SAS gains were adjusted to eliminate high frequency oscillation and to smooth out the rise and fall times in the angular rate response.

The yaw rate feedback was further adjusted after flight testing to improve controllability of the flight model. Since the Raptor has no vertical stabilizer, a large amount of yaw damping was required to smooth out oscillations caused by collective and pedal perturbations. Responses to a pedal step input with and without SAS are shown in Figure 11.

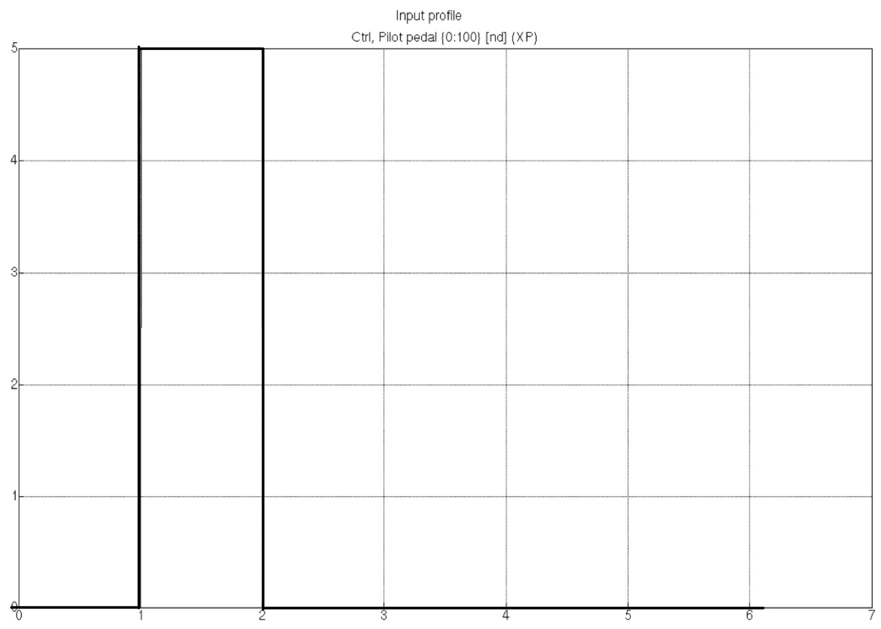

Figure 11 (a). Pedal step input

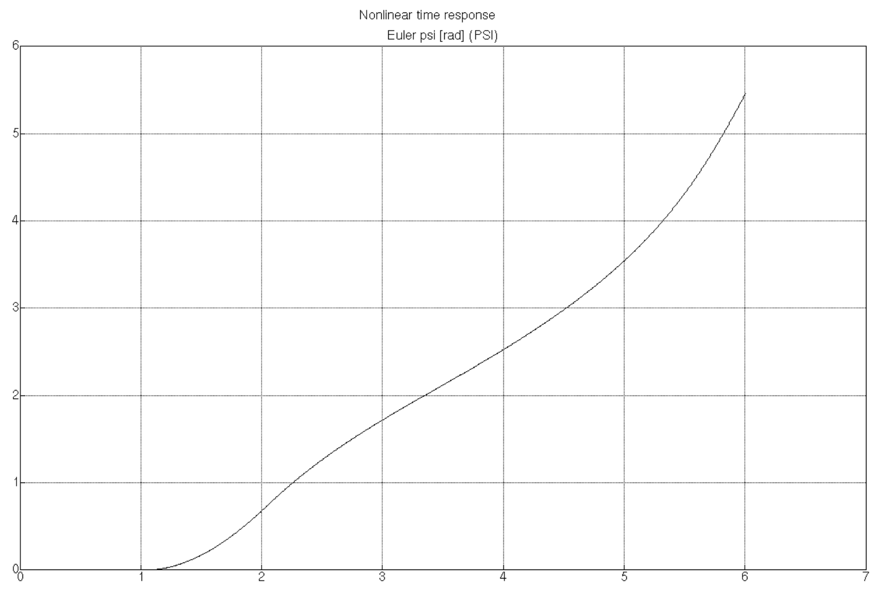

Figure 11 (b). Yaw response without SAS

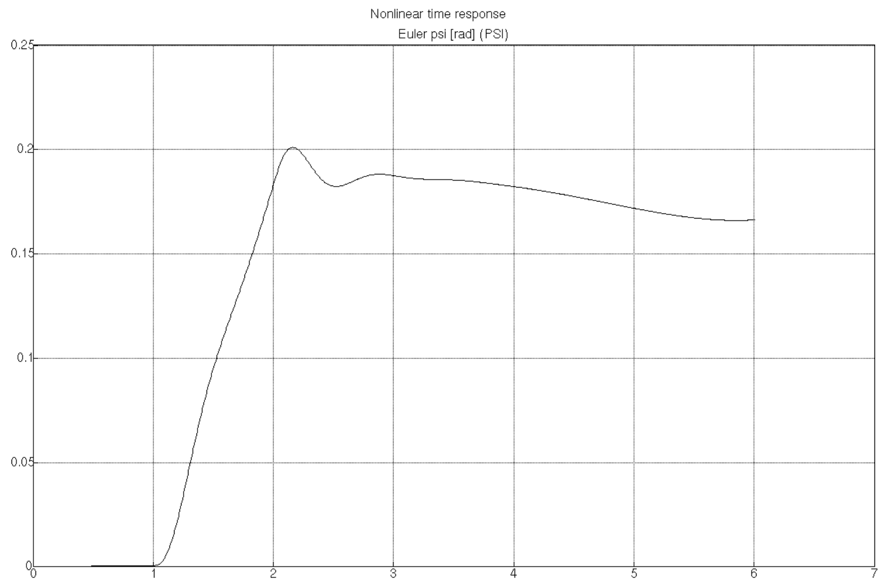

Figure 11 (c). Yaw response with SAS

Linkage Gains

The linkage gains for the control inputs were adjusted to give a response of a suitable magnitude. In the case of the collective input, a gain was selected to give a physically correct range of blade pitch motion as measured from the actual helicopter. Since the cyclic and pedal sensitivity of the Raptor were unknown, best guesses for the gains of these components were used. It was ensured that the cyclic gains produced responses of similar sensitivity in the lateral and longitudinal directions. The control system responsiveness may be refined in the future if handling data for the Raptor 90 is obtained.

Linkage Biases

Once the control system gains were determined, the biases were adjusted to obtain mid-range control inputs during trim in the hover condition. This was done by repeatedly trimming the helicopter, noting the control input deflections at trim, and adjusting the biases appropriately.

Yaw-Heave Mixing

A Nonlinear Response analysis was used to compare the induced yaw rate for a collective step input for a variety of mixing amounts, allowing an appropriate mixing factor to be determined.

5. Flight Model Simulation & Visualisation

5.1 Overview

When developing the flight model and control systems of an aircraft, it is useful to have the ability to simulate and visualise the aircraft's flight in real time. Simulation is particularly important for the development of a UAV due to the inherent complexity of autonomous flight control systems and the large scope for error. Through simulation and visualisation of a UAV prior to flight, flaws in critical systems can be identified and corrected without risk of crashing the aircraft.

5.2 Requirements

The simulation system used for the Raptor needed to have the ability to interface with MATLAB/Simulink; the development environment for the helicopter's high level flight control system. FLIGHTLAB provides a means of simulating a flight model in real time, however the integrated pilot visualisation module, PilotStation, is only of limited use as it is not easily integrated with other software such as MATLAB/Simulink. As an alternative to PilotStation, an open source flight simulator, FlightGear, was chosen as a means of visualising the FLIGHTLAB simulation.

5.3 FlightGear

The main reasons for the choice of FlightGear as a visualisation tool were:

Network Interface

FlightGear is able to communicate with external systems via a network interface [5]. This enables straightforward integration into an existing simulation framework.

Simulink Support

The Simulink aerospace blockset includes a module for transmitting directly to FlightGear via network, allowing a simulation in the MATLAB/Simulink environment to be visualised with little implementation work required.

High Level Features

Compared with a pure visualisation/rendering engine, FlightGear incorporates high level features one would expect for a flight simulator. This includes a global coordinate system, world terrain database, time of day and weather conditions, multiple camera views, and avionics displays.

Open Source

Being open source, FlightGear is extendable. This allows hard coded features to be directly added or modified as required, within the terms of the GPL license.

5.4 MATLAB/Simulink Integration

MATLAB/Simulink was used to develop a system model for handling the core update loop of the simulation. Raw control inputs are received from the joystick, scaled appropriately, and passed to the FLIGHTLAB runtime library as cyclic, collective and pedal control inputs. The aircraft's simulated position and attitude is then transmitted to FlightGear via the computer's network where the visual aircraft model is updated correspondingly. Figure 12 shows the process.

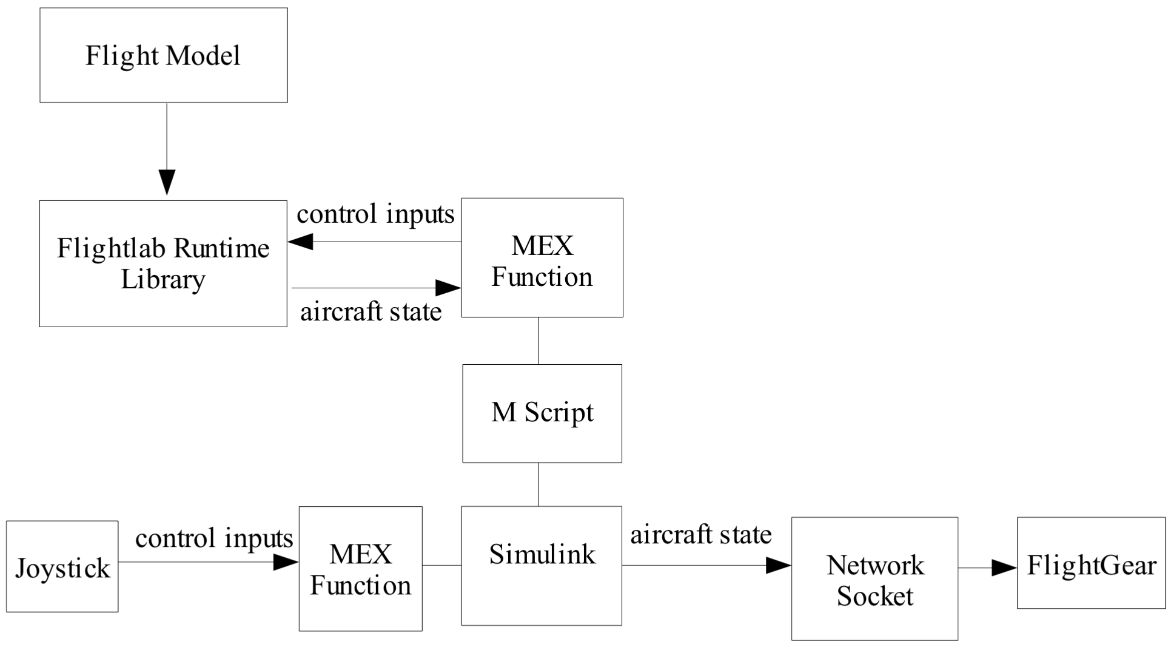

Figure 12. Flight model simulation & visualisation system modules

5.4.1 Control Inputs

To obtain control input data in Simulink, a joystick interface block was required. Although joystick input is supported for Simulink under MS Windows, the Linux version of Simulink has no built in joystick support. Since we were running FLIGHTLAB using the Linux operating system, a joystick interface needed to be developed.

Rather than read input device information directly from the operating system, an open source joystick library, libjsw 1.5.7, was used as a go-between. The library functions as a high level abstraction layer over the operating system's joystick interface, providing a simple, generic means of handling joystick input events.

A MATLAB/Simulink wrapper for the library was written in C and compiled into a MEX function. This function was then loaded into an S-Function block within Simulink, providing a graphical means to connect joystick axes inputs to other blocks in the system. Parameters such as device name, axis limits, and number of axes were specifically hard-coded into the MEX function for the Saitek X52 joystick used. The Simulink joystick block could be improved upon in the future by offering a user interface for configuring input devices, eliminating the need to recompile the MEX function if hardware is changed.

In addition to the joystick input, Simulink sliders were used in the system as supplementary input, allowing the aircraft to be trimmed or controlled when no joystick is connected. Finally, step function generators were connected to each slider, allowing the flight model response to specific step inputs to be observed and analysed.

5.4.2 FLIGHTLAB Library

One of the authors (Manso) has previously developed a MEX function for MATLAB which provides an interface for the FLIGHTLAB library. A MATLAB M script was used to pass control inputs to this MEX function and return aircraft motion data. Each call of the MEX function advances the FLIGHTLAB simulation by one time step.

The MEX function had originally been configured to work with the Super Seasprite flight dynamic model developed by Manso. Because the Raptor model is less complicated than this model, the FLIGHTLAB model export script required some modifications to be compatible with the MEX function. These modifications consisted of removal of references to certain variables in the FLIGHTLAB library which were not supported by the Raptor model.

Due to the fact that FLIGHTLAB operates in Cartesian 3D space while FlightGear uses a curved earth model, a conversion of position coordinates was required. Simulink's ‘Flat Earth to Lat/Long’ block was used to convert Cartesian coordinates to latitude, longitude and altitude. A reference latitude, longitude and altitude were used to specify the location of the origin of the Cartesian coordinate set.

The Simulink time step frequency was set to correspond to the update frequency required by FLIGHTLAB to run the model in real time. By default, Simulink runs the simulation faster than real time, advancing at the maximum rate allowed by CPU performance. In order to synchronise Simulink's pace with FlightGear, the 'Set Pace' block is used. this allows the ratio between simulation seconds and Simulink clock seconds to be specified.

5.4.3 Simulink - FlightGear Interface

Simulink includes an existing interface block for transmitting aircraft state data to FlightGear. Aircraft parameters are bundled into a data packet which is then sent to a designated network socket using UDP protocol. FlightGear must be specifically configured to read aircraft state data from this socket rather than attempting to simulate the aircraft internally. This is done by starting FlightGear with the appropriate set of command line parameters, within which the network IP address and port is specified. The parameters used for testing the Raptor model were:

```
fgfs --aircraft=ec135 –fdm=network,localhost,5501,5502,5503 --timeofday=noon
```

In addition to specifying command line parameters, the aircraft model within FlightGear must be made network enabled by editing its XML setup file and setting the 'flight-model' parameter to 'network'.

5.5 Testing

The Simulink-FLIGHTLAB system was observed to operate correctly, and the Raptor FLIGHTLAB model was successfully flown in real time under joystick control. The complete simulink system is shown in Figure 14.

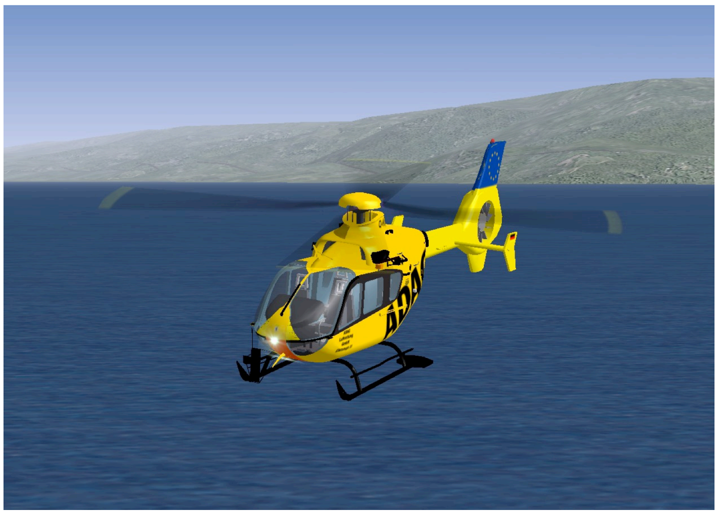

Figure 13. Screenshot from visualisation system

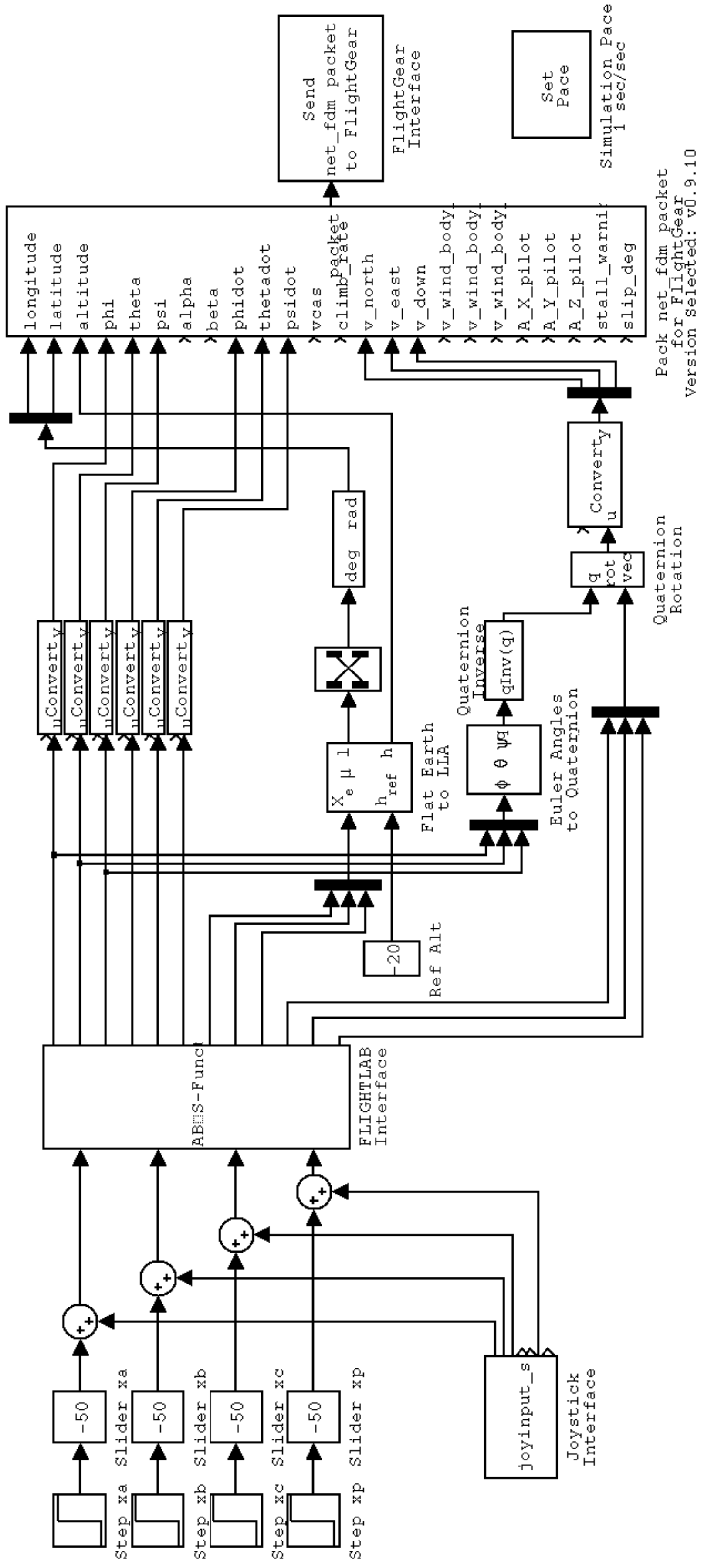

Figure 14. Complete Simulink system

6. Inertial Motion Unit Visualisation

6.1 Overview

In order for an UAV to fly in a controlled manner and navigate autonomously, it must be capable of sensing its position, attitude, and motion at any instant. In the case of the Raptor UAV, this capability will be provided by an Inertial Motion Unit (IMU). An IMU uses inertial sensors to derive the acceleration, velocity, position, orientation and angular rates of the unit.

IMUs have a tendency to drift over time, that is, the deviation between the actual position of the unit and the internally derived position increases. In addition, the output signal from an IMU may be noisy when approaching the limits of the unit's measurement resolution. When integrating an IMU into a flight control system, it is often useful to have a means of visualising the raw data output of the IMU so that techniques for drift compensation and signal filtering can be evaluated more effectively.

6.2 Implementation

An interface for the Crossbow NAV420CA IMU was developed and integrated into the existing visualisation system. Although the NAV420CA is a larger unit than that which would be used on board the Raptor, its features are representative of a typical aircraft mounted IMU.

Figure 15. Crossbow NAV420CA inertial measurement unit

The NAV420CA communicates via a DB-9 serial connector using the RS232 standard, allowing the unit to interface with a PC serial port. Data is transmitted and received in discrete packets, with each packet corresponding to a certain IMU function such as sending sensor data or changing an internal setting. A code module for creating, sending, receiving, and interpreting packets was written in C using standard Unix serial interface libraries.

The following IMU packets were implemented:

Outgoing

- set packet rate
- set baud rate
- set packet mode – scaled sensor data, angle, NAV
- send request for packet
- send ping

Incoming

- Ping
- Scaled sensor data
- Angle data
- NAV data

The serial interface C code was compiled as a MEX function, enabling the IMU to be accessible via MATLAB. A Simulink block was then written in M script to interface with the MEX function, making IMU sensor outputs accessible from the visualisation system.

6.3 Testing

The IMU interface was tested by transmitting motion data directly to a FlightGear aircraft in real time. The aircraft's attitude was observed to correlate well with IMU orientation. The acceleration and velocity data from the IMU was not readily testable due to space restrictions in the test area. Future tests involving a vehicle mounted IMU with GPS input may be possible to obtain a full 6 degrees-of-freedom motion visualisation.

7. Autopilot System

7.1 Overview

A key requirement of the Raptor UAV is that it must be capable of flying autonomously. To investigate the potential of the Raptor flight model for autonomous flight, an autopilot system was developed using MATLAB/Simulink. This system serves as a proof of concept and may act as a framework for future autopilot systems.

7.2 Research into existing systems

Initially, it was hoped that existing open source UAV autopilot software could be utilized in the Raptor autopilot system. Of the open source autopilot systems investigated (listed in Appendix C), none were found to be advanced or extendable enough for what the Raptor UAV required. It was therefore decided that a custom system should be developed.

7.3 Autopilot Design

The autopilot system developed for the Raptor UAV (Figure 16) is split into two sections; a high level navigation system and low level control system. The function of the navigation system is to take general flight plan information and generate a desired flight path. The low level autopilot control system is responsible for controlling helicopter attitude and thrust in such a way as to maintain the desired flight path.

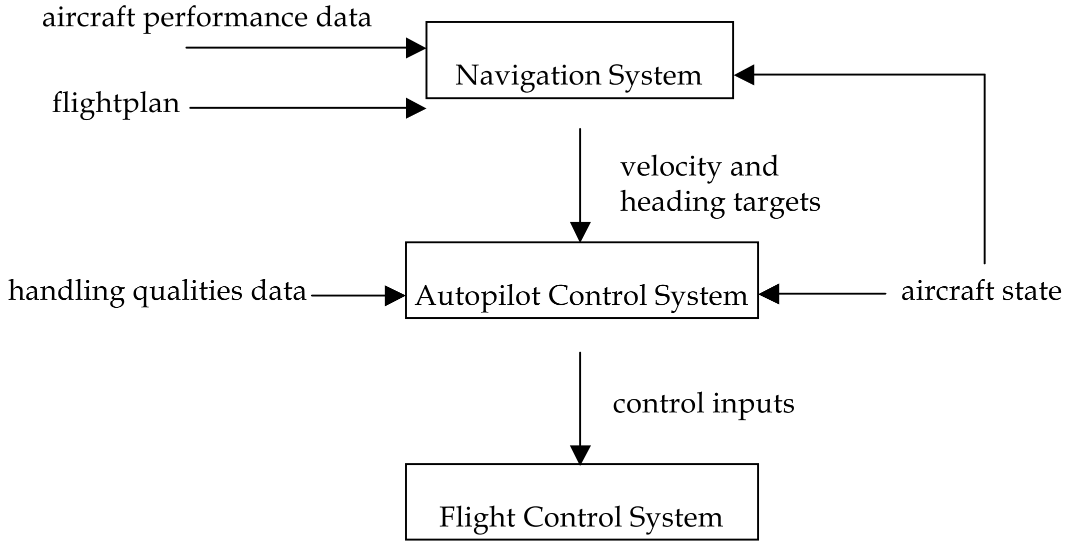

Figure 16. Autopilot system overview

7.3.1 Navigation System

The navigation system tracks the helicopter's position in relation to an assigned flight plan and advances from one waypoint to the next when appropriate. A flight plan consists of waypoints and interconnecting legs, as shown in Figure 17. Each waypoint and leg may be assigned properties such as speed and heading, allowing various manoeuvres to be defined. These include takeoff, landing, hovering, forward flight, and low speed backward and sideways flight.

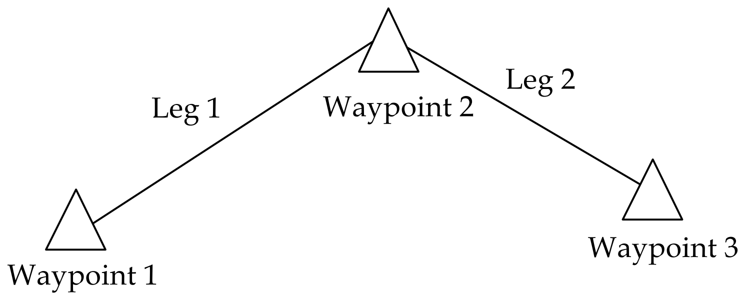

Figure 17. Waypoints and interconnecting legs

Waypoint properties:

Position
Location of the waypoint in 3d space.

Flythrough speed (optional)
Speed of the aircraft when it arrives at the waypoint. When not specified, the speed of the next leg is used instead.

Hold flag
When enabled, the helicopter will maintain a hover at the waypoint.

Heading (optional)
Heading to maintain when hovering at a waypoint. Only used when hold flag is set.

Leg properties:

Speed
Speed to maintain when flying along leg.

Heading (optional)
When a leg heading is specified, the helicopter will adjust its forward and lateral velocity as needed to maintain its flight path, while holding the assigned heading. The ability to hold a constant heading irrespective of flight path may have operational uses such as keeping sensors or weapons oriented toward a target while performing tactical manoeuvres. When this parameter is not specified, the helicopter will travel to the next waypoint in conventional forward flight.

7.3.2 Autopilot Control System

The autopilot control system is responsible for generating a set of cyclic, collective and pedal inputs which will allow the helicopter to follow a desired flightpath. A key requirement of this system is that it must converge upon optimum control inputs in a short amount of time and with minimum oscillation. The system should also be capable of correctly handling worst case attitude perturbations; for example the helicopter must be capable of righting itself if flight becomes inverted.

Attitude Control

The attitude control system consists of several proportional controllers in series, with the output of one connected to the input of the next, as shown in Figure 18. The desired pitch and roll attitudes are determined directly from the difference in the desired velocity and the actual velocity of the helicopter. Because the Raptor requires a slight non-level attitude when holding a fixed ground position, there is some steady-state error associated with this technique. The error is relatively small however, and could be eliminated in the future by determining desired attitude with a higher order system.

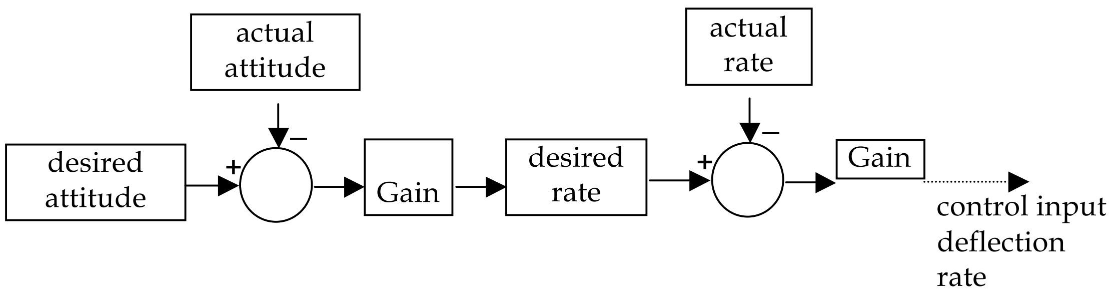

Figure 18. Attitude control via proportional controllers in series

Appropriate control inputs for attaining desired angular rates are determined by adjusting the control deflection rate rather than setting the absolute control deflection. This is to allow the helicopter to be flown in flight conditions where non-zero control deflections are required to achieve trim. Damping is applied to obtain control deflection convergence.

Climb Control

Collective control inputs are determined in a similar manner to cyclic and pedal inputs. The collective control input deflection rate is set proportionally to the difference between desired and actual climb rate, and damping is applied.

7.4 Testing

The autopilot system was tested using the Raptor flight model in the simulation system developed previously. The helicopter was able to follow a flight plan as expected, performing manoeuvres with a high degree of stability. A small amount of oscillation and steady state error was observed, particularly in heading and velocity control. It is anticipated that these issues may be resolved simply by adjusting control system parameters for gain and damping.

The autopilot system may also benefit from the implementation of Kalman filtering, which can be programmed via a neural network to compensate for cross-coupling effects. Compared with proportional controllers alone, Kalman filtering may provide faster control convergence with less oscillation, especially in adverse wind conditions [6].

8. Conclusions

While the accuracy of the Raptor flight model is currently unknown, the model's performance and handling during flight testing appears to be qualitatively appropriate. The accuracy of the flight model would be improved significantly if experimental data from flight testing of the physical aircraft becomes available.

The real time simulation system allowed the Raptor flight model to be flown successfully under pilot control and autonomous control. The system's autopilot module proved to be capable in flying the helicopter along a predetermined flight path. Although the control system component of the autopilot was found to produce stable flight, the system suffers from minor oscillation and steady state error in velocity and heading control. These problems may be resolved in the future through the implementation of higher fidelity controllers, possibly utilizing Kalman filtering to compensate for cross-coupling effects.

The IMU interface developed for MATLAB/Simulink was found to function as expected, allowing state data from the IMU to be visualised in FlightGear via Simulink. An appropriate test environment is required before IMU position and velocity can be visualised.

9. Acknowledgements

The work performed under the SVS was carried out by Mr Matthew Reid under the guidance and support of Mr Sylvain Manso. The authors wish to thank Mr. William Oliver for offering additional technical support during software installation and operation of the Inertial Motion Unit. For providing information on the Raptor helicopter and assisting with measurement taking, thanks go to Mr. John Yildiz.

10. References

1. Garcia, R. D., Valavanis, K. P., Kontitsis M., *A High Power, Inexpensive On-board Vision System for Miniature Unmanned VTOL Vehicles*, University of South Florida, 2005
2. Kellets Hobbies 2007, *CASA Thunder Tiger – Raptor 90 SE,* viewed 7 February 2008, <http://www.kellettshobbies.com.au/Home/tabid/36/CategoryID/24/List/1/Level/a/ProductID/11/Default.aspx>
3. Frye, M. T., Qian, C, Colgren, R. D., *Receding Horizon Control of a 6-DOF Model of the Raptor 50 Helicopter: Robustness to Changing Flight Conditions*, 2005 IEEE Conference on Control Applications, Toronto, Canada, 2005
4. Mettler, B., Kanade, T., Tischler, M. B, *System Identification Modeling of a Model-Scale Helicopter*, Carnegie Mellon University, 2003
5. FlightGear Flight Simulator 2006, *FlightGear features*, viewed 7 February 2008, <http://FlightGear.org/features.html>
6. Autopilot Sourceforge.net Page 2005, Kalman Filtering, viewed 7 February 2008, <http://autopilot.sourceforge.net/kalman.html>
7. RunRyder Discussion Forum 2002, Thread: 'Anyone used a GPS to measure their heli's top speed?', viewed 7 February 2008, <http://www.runryder.com/helicopter/t24301p1/>

Appendix A: Flight Model Parameters

A.1 Solution Parameters

The number of rotor azimuth steps/rev was set to 7 to give a simulation update rate of 180 hertz.

A.2 Environment

The default environmental conditions provided by FLIGHTLAB were used.

A.3 Main Rotor – Disk Main Rotor Model

Parameter	Value	Source
Rotor Direction	Counter clockwise	Investigation of physical helicopter shows rotors spin clockwise. Set to counter clockwise to overcome FLIGHTLAB error
Hub location	0, 0, 110 mm	Physically measured
Number of Blades	2	Observed
Radius	1557.2mm	Physically measured
Blade tip loss factor	0.97	FLIGHTLAB default
Lift coefficient at zero AoA	0	FLIGHTLAB default
Lift curve slope	5.73	FLIGHTLAB default
Airfoil drag coefficient	0.0087	FLIGHTLAB default
Aerodynamic Root Cutout	401 mm	Physically measured
Solidity weighted chord	130 mm	Calculated using formula $\frac{\pi\sigma R}{N}$, where σ is the solidity, R is the radius and N is the number of blades
Linear blade Twist	0.1°	Physically observed to be 0°, but set to 0.1° to overcome FLIGHTLAB error
Flap hinge offset	267 mm	Arbitrarily set as a third of blade radius
Mass	0.181 kg	Physically measured
Flapping Moment of inertia	0.03775 kg m^2	Calculated using moment of inertia formula $\frac{MR^2}{3}$
Flapping frequency	1.0358	FLIGHTLAB default
Nominal Rotor Speed	1550 RPM	Raptor specifications
Swashplate phase angle (offset)	-15°	Chosen experientially for minimum cross coupling

A.4 Tail Rotor – Bailey Rotor Model

Parameter	Value	Source
Rotor Hub Location	926, 55, -73 mm	Physically measured
Orientation in Euler axes	0, 0, -90°	Assumed
No. Of blades	2	Photos, RMAX website
Radius	134.7	Physically measured
Tip loss factor	0.92	FLIGHTLAB default
Lift curve slope	5.73	FLIGHTLAB default
Rotor head drag coefficient	0	FLIGHTLAB default
Airfoil constant drag coefficient	0.0087	FLIGHTLAB default
Airfoil 1st order drag polar constant	-0.0216	FLIGHTLAB default
Airfoil 2nd order drag polar constant	0.4	FLIGHTLAB default
Pitch bias	0°	Assumed
Solidity weighted blade chord	20.95 m	Calculated using formula $\frac{\pi\sigma R}{N}$, where σ is the solidity, R is the radius and N is the number of blades
Linear blade twist	0.1°	Measured to be 0°, but set to 0.1° to overcome FLIGHTLAB error
2nd moment of inertia	4.2 kg m^2	
Tangent of delta three	0.7002075	FLIGHTLAB default
Partial of coning w.r.t thrust	0.001455 deg/lbf	FLIGHTLAB default
Initial collective pitch setting	0	Assumed
Blocking effect for low speed	0.796	FLIGHTLAB default
Speed threshold for blocking effect.	100 knots	Arbitrarily set to prevent blocking. Raptor aerodynamic properties are currently unknown
Nominal Rotor Speed	7170 RPM	Derived from main rotor RPM and measured gear ratio

A.5 Airframe

Fuselage – Rigid Fuselage

Parameter	Value	Source
C.G. location	0, 0, -70 mm	Physically Measured
Mass	4.8 kg	Physically Measured
I_{xx}, I_{yy}, I_{zz}	0.0769, 0.1973, 0.1912 slug-ft^2	Fryer et al
I_{xy}, I_{xz}, I_{yz}	0	Assumed negligible

Fuselage Airloads

Parameter	Value	Source
Characteristic dimension	1411mm	Assumed as Raptor length
Reference area	0.11 m^2	Determined experientially
Reference length	1411mm	Assumed as Raptor length

A.6 Landing Gear

Parameter	Value	Source
Left front location Right front location Left rear location Right rear location	-215, 130, -205 mm 531, -354, 354 mm 1132.7, 354, 354 mm 1132.7, -354, 354 mm	Physically measured
Maximum gear reaction load	-50 N	Estimated from Raptor mass
Undeflected length	100 mm	Physically measured
Damping coefficients	200 200 kg/s	Scaled from FLIGHTLAB defaults
Stiffness coefficients	2000 2000 kg/sec^2	Scaled from FLIGHTLAB defaults
First stage max compression	50 mm	Exaggerated estimated for stability
Transition velocity for friction coefficient	0.01 m/s	Arbitrarily set to small stable value
Weight on wheel force	0.001 lbf	Arbitrarily set to small stable value
Effectively tire radius	10 mm	Arbitrarily set to small stable value

A.7 Propulsion – Ideal Engine Model

Parameter	Value	Source
Number of engines	1	Physically observed
Nominal engine torque	1.55 Nm	Calculated using formula $\tau = \frac{\dot{W}}{\omega}$, where $\dot{W}$ is power, and ω is angular velocity.

Appendix B: Data Sources for Raptor Top Speed Estimation

Results obtained from RunRyder internet forum [7].

Helicopter	**Relation to Raptor 90**	**Top Speed**
Raptor 50	Raptor 90 is of similar size to Raptor 50 and has a more powerful engine	58 knots
Raptor 30	Raptor 90 is larger than Raptor 30 and has a more powerful engine	52 knots

Appendix C: Open Source Autopilot Systems

Name	**Features**	**Language**	**URL**
Autopilot: DIY UAV	- 2 DOF proportional controller - Kalman filtering	C	http://autopilot.sourceforge.net
Paperazzi	- 3 DOF proprtional controller - Basic navigation	OCaml	http://paparazzi.enac.fr
JSBSim	- Basic control system blockset - Sample autopilot implementations	C++	http://jsbsim.sourceforge.net

Page classification: UNCLASSIFIED

DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION **DOCUMENT CONTROL DATA**	1. DLM/CAVEAT (OF DOCUMENT)
2. TITLE Development of a Rotary Wing Unmanned Aerial Vehicle (UAV) Simulation Model	3. SECURITY CLASSIFICATION (FOR UNCLASSIFIED REPORTS THAT ARE LIMITED RELEASE USE (L) NEXT TO DOCUMENT CLASSIFICATION) Document (U) Title (U) Abstract (U)
4. AUTHOR(S) Matthew Reid and Sylvain Manso	5. CORPORATE AUTHOR DSTO 506 Lorimer St Fishermans Bend Victoria 3207 Australia

6a. DSTO NUMBER DSTO-GD-0791	6b. AR NUMBER AR-015-877	6c. TYPE OF REPORT General Document	7. DOCUMENT DATE March 2014

8. FILE NUMBER	9. TASK NUMBER	10. TASK SPONSOR	11. NO. OF PAGES 30	12. NO. OF REFERENCES 7

13. DSTO Publications Repository http://dspace.dsto.defence.gov.au/dspace/	14. RELEASE AUTHORITY Chief AD (Aerospace Division)

15. SECONDARY RELEASE STATEMENT OF THIS DOCUMENT

Approved for public release

OVERSEAS ENQUIRIES OUTSIDE STATED LIMITATIONS SHOULD BE REFERRED THROUGH DOCUMENT EXCHANGE, PO BOX 1500, EDINBURGH, SA 5111

16. DELIBERATE ANNOUNCEMENT

No Limitations

17. CITATION IN OTHER DOCUMENTS Yes

18. DSTO RESEARCH LIBRARY THESAURUS

FLIGHTLAB, UAV, helicopter, simulation

19. ABSTRACT

This report details the work undertaken during a DSTO Summer Vacation Scholarship Program on the development of a flight simulation system for the Raptor 90 UAV platform. Simulink was used to execute the core simulation update loop, with externally linked modules providing functionality for joystick input, flight dynamics, and 3D visualisation. A flight model for the Raptor was developed in FLIGHTLAB, and flown in the simulation system under pilot control and autonomous control. The Raptor model was shown to be capable of flying autonomously through a range of manoeuvres, via the use of an autopilot code module written in MATLAB. Additionally, a MATLAB interface for an inertial motion unit was developed to aid in future autopilot research.

Page classification: UNCLASSIFIED